ARBITRATED LOOP PORT SWITCHING

TECHNICAL FIELD

The present invention relates to routing data between various devices such as may be interconnected through hubs, storage area networks, networked attached storage, and the like.

BACKGROUND ART

Large computer storage systems comprise arrays of disk and tape drives with several controllers directing the flow of data between the disk drives and the computers. A common network topology is to have drives linked together by two linear unidirectional communication busses running in opposite directions with a controller at each end. This approach allows both controllers to communicate with each drive in the array. In practice, the workload between the controllers is often divided so that one controller services only a subset of the drives, while the other controller services another, possibly overlapping, subset of drives.

The performance of such an arrangement varies, in part, with the topology of the interconnection network. For example, each controller has a direct link to the first drive immediately adjacent on the bus. Since no other controller communicates across this segment of the bus, the full bandwidth of the bus is available between the controller and the first adjacent drive. However, as the controller tries to communicate with drives further away on the network, the controller may come into contention with other controllers trying to communicate with other drives causing an effective reduction in the available bandwidth. Another limitation of this approach is that the network relies on the continued operation of all drives at all times to keep the busses operational. When a drive

fails or loses power, the busses are broken at that point, isolating the controllers from drives on the far side of the failed device.

Another network topology replaces the linear unidirectional communication buses with two communication loops. In this topology there is one controller per loop. This approach also allows both controllers to access all of the disk drives in the array, and eliminates contentions between controllers by isolating them on separate loops. This approach has a practical limitation in that each disk drive must have one loop interface for each communication loop to which it is tied. As the number of controllers on dedicated communication loops increases, the cost of each drive increases due to the increase in the number of interfaces it must support. This approach may also be susceptible to failed nodes disrupting the network. If the network technology cannot route the messages through an unpowered or failed node, then the loop is broken at that point, preventing controller access to the disk drives further around the loop.

In a Fibre Channel (FC) network, devices such as controllers and drives are connected by a network or arbitrated loop. Only two of the devices may communicate over a loop or point-to-point network at one time. Other devices must wait to communicate. The system latency of a Fibre Channel arbitrated loop may be reduced by subdividing the network into multiple subloops. Each subloop can operate independently thus allowing for one message transfer operation to occur simultaneously within each of the subloops. Applying this approach to disk arrays, one controller and the disk drives it services most can be assigned to each subloop. When the controller in one subloop places a request to communicate with a drive in another subloop, a hub links the two subloops. Linking the subloops through the hub allows any controller to reach any drive in the array. However, the hub only supports one source-to-destination inter-subloop link at a time. Further, while the inter-subloop link is established, the controllers in the source and destination subloops must arbitrate with each

other. As with other loop topologies, a failed or unpowered node in a subloop may disrupt that subloop.

Hubs have been used to eliminate loop topology vulnerability to a break in the loop. The hubs physically connects to each of the nodes in a star type arrangement with connections radiating out from the hub. If a node fails or loses power, the hub circuitry senses the loss of message traffic to and from the node and switches out the failed node. Individual drives and controllers can be switched out due to failure or for maintenance and repair without a major disruption to the rest of the network. Further, new devices can be switched into the loop while the network is operational. Limitations with this approach include high cost and the need to share network bandwidth with all controllers competing for use of the network.

DISCLOSURE OF INVENTION

The present invention connects a group of nodes, such as controllers and drives, onto a separate or private communication loop so that the group of nodes accesses the full bandwidth of the private communication loop. The apparatus and method of the present invention are adaptable to any arbitrated loop network. In an embodiment, a hub operates from a subset of the messages defined in the Fibre Channel arbitrated loop protocol while allowing each node to use the full protocol.

The present invention provides for a hub interconnecting a plurality of nodes, each node having a channel over which data is transmitted and received. The hub includes a port interface in communication with each node through the channel. Each port sends data over a send data path and receives data over a receive data path. An interconnect device forwards data between any send data path and any receive data path. A controller signals the interconnect device to form at least one separate communication loop including at least two of the nodes.

The controller may form each separate communication loop based on a message received from at least one port included in the separate communication loop.

A method of interconnecting a plurality of nodes is also provided. A request is received from a first node to access a second node. A determination is made as to whether or not the second node is not busy. If the second node is not busy, a separate communication loop is formed including the first node and the second node.

In an embodiment, a request is received from the first node to access a third node. If the third node is not busy, the third node is joined in the separate loop including the first node and the second node.

The present invention also provides a switching hub connected to each node of a multiple node network with a sending channel and a receiving channel. The switching hub has an interconnect switch capable of connecting the sending channel and the receiving channel of each node into a separate communication loop. The switching hub has a plurality of port interfaces, each port interface linking the respective receiving channel and the respective sending channel of each node to the interconnect switch. Each port interface detects messages on the receiving channel. A controller controls the interconnect switch to form at least one separate communication loop based on at least one detected message.

In an embodiment of the present invention, the plurality of nodes communicate with each other using a protocol defining message types including Arbitration, which contains at least a source addresses, Open, which contains at least a source address and a destination address, and Close.

In another embodiment of the present invention, the controller forms a separate communication loop connecting a first node and a second node,

the first node requesting access to the second node. The controller may also form the separate communication loop connecting a third node requested by the first node. The controller may also form the separate communication loop connecting a fourth node requesting access to the second node.

A method for controlling multiple nodes on a network is also provided. A first node places an access request and is connected to a separate communication loop. The first node requests a message transfer operation with a second node. The second node is added to the separate communication loop. When connection is no longer required, a close message is received. The nodes may be released from the separate communication loop if its status is not busy after a waiting period times-out.

These and other objects, features and advantages will be readily apparent upon consideration of the following detailed description in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF DRAWINGS

FIGURE 1 is an electrical block diagram of the switching hub according to an embodiment of the present invention;

FIGURE 2 is a logic flow diagram implemented by the switching hub to create and eliminate a private communication loop according to an embodiment of the present invention;

FIGURE 3 is a block diagram of the network after initialization with one main communication loop according to an embodiment of the present invention;

FIGURE 4 is a block diagram of the network with a first disk controller switched to a first private communication loop according to an embodiment of the present invention;

FIGURE 5 is a block diagram of the network with the first disk controller communicating with a first disk drive on the first private communication loop according to an embodiment of the present invention;

FIGURE 6 is a block diagram of the network with the first disk controller communicating with first and second disk drives on the first private communication loop according to an embodiment of the present invention;

FIGURE 7 is a block diagram of the network after the first disk drive has been returned to the main communication loop according to an embodiment of the present invention;

FIGURE 8 is a block diagram of the network with the first disk controller communicating with the first disk drive on the first private communication loop, and the second disk controller switched to a second private communication loop according to an embodiment of the present invention;

FIGURE 9 is a block diagram of the network with the first disk controller communicating with the first disk drive on a first private communication loop, and the second disk controller communicating with the second disk drive on the second private communication loop according to an embodiment of the present invention; and

FIGURE 10 is a block diagram of the network with the first disk controller and second disk controller communicating with the first disk drive on the first private communication loop according to an embodiment of the present invention.

BEST MODE FOR CARRYING OUT THE INVENTION

Referring to Figure 1, an embodiment of the invention where switching hub **11** connects multiple nodes **30a-30n** using the Fibre Channel arbitrated loop protocol is shown. Each node **30a-30n** may be an input/output controllers and drives, such as may be used for magnetic or optical tapes or disks, processors, communication interfaces, monitoring and debugging equipment, and the like. Each of the nodes **30a-30n** communicates with port interfaces **12a-12n**, respectively, over a channel, such as a pair of sending channels **32a-32n** and receiving channels **34a-34n**. Each port interface **12a-12n** also connects with interconnect switch **20** over receive data path **22a-22n** and send data path **24a-24n** respectively. Interconnect switch **20** can transfer data from any receive data path **22a-22n** to any send data path **24a-24n** under the control of controller **26**.

Port interface **12a** illustrates the internal detail of a typical port interface. A receiver **14** converts the serial stream of signals arriving on receiving channel **32a** into parallel data, typically ten or twenty bits wide. The receiver **14** may also perform 8b/10b decoding of the data before passing the data to decoder **16**. The decoder **16** examines each message to determine the message type and, where appropriate, the message source address and destination address. The decoder **16** may buffer messages or send messages directly to interconnect switch **20** through receive data path **22a**. Messages leaving the interconnect switch **20** on send data path **24a** flow to a multiplexer **17**. The multiplexer **17** may also decode messages and notify the controller **26** of the message types, source addresses and destination addresses. The multiplexer **17** then passes the messages along to a transmitter **18**. The transmitter **18** perform the 8b/10b encoding of the data and converts parallel data into a serial stream of signals which are transmitted across the sending channel **32a** to the node **30a**. The multiplexer **17** can transfer messages to the transmitter**18** from the sent data path **24a** or from the controller **26**.

Initially, each port interface **12a-12n** is set to send Idle until a frame is received from an attached node **30a-30n**. When decoder **16** of port interface **12a** detects an Arbitration message, it notifies the controller **26** of the message type and the source address. Controller **26** may then set interconnect switch **20** to route information from receive path **22a** to send data path **24a**, effectively placing node **30a** on its own private loop. When decoder **16** then receives an Open message, decoder **16** notifies the controller **26** of the message type, the source address and the destination address, and buffers the Open message until commanded by the controller **26** to release it to the interconnect switch **20**. When the message type is Close, the decoder **16** notifies the controller **26** of the message type and passes the message along to the interconnect switch **20**.

The interconnect switch **20** forms data paths between the nodes **30a-30n**. A convenient initialization configuration is to daisy chain each of the receive data paths **22a-22n** to the adjacent send data paths **24a-24n** to connect all of the nodes **30a-30n** together as one main communication loop. Receive data path **22a** is connected to send data path **24b**, receive data path **22b** is connected to send data path **24c**, receive data path **22c** is connected to send data path **24d**, and so on until receiving data path **22n** is connected to send data path **24a**. The interconnect switch **20** can then switch the connections in response to commands from the controller **26** to create one or more separate, private communication loops with one or more nodes per loop. Interconnect switch **20** may be any device capable of forming multiple point-to-point logical connections such as, for example, one or more cross-point switches, cross-bar switches, routers, multiplexors, and the like.

The controller **26** is composed of a processor **27**, a busy port store **28**, and a valid arbitration loop address store **29**. Stores **28**, **29** may be implemented using a content addressable memory (CAM), a look-up table, or any other suitable means. The busy port store **28** is used by the processor **27** for very

fast address comparisons to determine the busy/not busy status of the nodes. Nodes actively involved in message transfer operations are "busy", those that are not are "not busy." The valid arbitration loop address store **29** is used by the processor **27** for high speed storage and comparison of message types and communication loop configurations. The processor **27** makes logic decisions and issues commands to the interconnect switch to set up, modify and take down private communications loops. The processor **27** also generates Busy and the Idle messages to notify the nodes **30a-30n** of various events during the establishment of private communication loops.

In an embodiment of the present invention, messages used by the switching hub **11** and the nodes **30a-30n** are only a subset of what is defined by the Fibre Channel arbitrated loop protocol. This avoids the need for complicated and expensive circuitry in the switching hub **11** hardware, and requires no special modifications of the nodes **30a-30n**. Also, the switching hub **11** transfers all of the messages between the nodes **30a-30n** without modification, except for the Open message which is only delayed until the private loop is established. This makes the switching hub **11** virtually transparent to the nodes **30a-30n**. Thus, the present invention supports other Fibre Channel classes of service at no additional cost. The transparent message transportation allows the present invention to support other protocols such as Small Computer System Interface (SCSI), Internet Protocol (IP), and the like. The present invention also allows for various media types including fiber optics, fiber optics, coax, triax, twisted shielded pair, and the like, by substituting appropriate transmitter **18** and receiver **14** circuits.

Referring now to Figure 2, a flow diagram illustrating hub logic to create and eliminate a private communication loop according to an embodiment of the present invention is shown. As will be appreciated by one of ordinary skill in the art, the operations illustrated are not necessarily sequential operations. Similarly, operations may be performed by software, hardware, or a combination

of both. The present invention transcends any particular implementation and aspects are shown in sequential flow chart form for ease of illustration.

The starting network configuration, for example, has all of the nodes **30a-30n** connected in one main communication loop. Node A, such as node **30a** in Figure 1, requests to use the network by transmitting an Arbitration message. Upon detection of the Arbitration message, as in block **40**, the switching hub **11** switches nodes A from the main communication loop to a newly created private communication loop, as indicated by block **42**. Node A knows it has won arbitration when it receives back its own Arbitration message. After winning arbitration, nodes A transmits an Open message containing the destination address of another node, for example node B, such as node **30b** in Figure 1. When the switching hub **11** detects the Open message, as indicated by block **44**, it buffers the message and sends an Idle message back to node A as acknowledgment, as in block **46**. The switching hub **11** then checks the busy/not busy status of node B, as in block **48**. If node B is busy, the switching hub **11** sends a Busy message to node A to notify it that node B is busy and thus not available, as in block **50**. When the status of node B is not busy, the switching hub **11** switches, node B to the private loop, as in block **52**. The status of both node A and node B is changed to busy, as indicated by block **54**. The buffered Open message to B is released, as indicated by block **56**. This Open message propagates out on sending channel **34b** to node B to inform node B that node A is trying to communicate with it. Node B returns the Open message on the receiving channel **32b**, back though the switching hub **11** and out to node A on sending channel **34a**. When node A receives the Open message it knows that node B is available for communication. The switching hub **11** then facilitates the message transfer operations between node A and node B, as in block **58**.

When node A and node B are finished transferring messages, one or both will issue a Close message. When the switching hub **11** detects the Close message, as in block **60**, it changes the status of node A and node B to not busy,

as in block **62**. Hub **11** waits a predetermined amount of time to see if node A and node B will start another message transfer operation or not, as in block **64**. After the wait indicated by block **64** times-out, the busy/not busy status of node A is checked, as indicated by block **66**. If node A is busy with another message transfer operation then the switching hub **11** leaves node A on the private communication loop. If node A is not busy, then it is switched back to the main communication loop, as indicated by block **68**. Similarly, after the wait, indicated by block **64**, the busy/not busy status of node B is checked, as indicated by block **70**. If node B is busy with another message transfer operation then the switching hub **11** leaves node B on the private communication loop. If node B is not busy, then it is switched back to the main communication loop, as indicated by block **72**.

The wait indicated by block **64** allows the switching hub to operate more efficiently by avoiding the need to return the nodes to the main communication loop after each message transfer. It also allows node A and node B to request the addition of a third node to the private communication loop. This feature is useful, for example, in an embodiment where node A is a first disk controller, node B is first disk drive, and the first disk controller wishes to talk to a second disk drive. By pulling the second disk drive onto the private communication loop, the first disk controller can time share the loop bandwidth between the two disk drives in any ratio the first disk controller requires.

A third node, for example a second disk controller wishing to talk briefly to the first disk drive, can also request to be added to the private communication loop without forcing the first disk controller back into the main loop. The first and second disk controllers can then arbitrate between themselves for the appropriate amount of loop bandwidth. If the second disk controller has a priority higher than the first disk controller, the first disk controller will remain in a not busy state for a predetermined amount of time, as indicated by block **64**. The first disk controller can then be switched back to the main communication

loop, as indicated by block **68**, leaving the second disk controller with the full bandwidth of the private communication loop.

Referring now to Figures 3 through 9, a series of examples of the operation of the preferred embodiment of the present invention linking a set of disk controllers **C1-C3** with an array of disk or tape drives **D1-D5** is shown. While controllers and drives are shown, it will be understood by one of ordinary skill in the art that controllers **C1-C3** and drives **D1-D5** may be any type of node. The internal circuitry of the switching hub **11** is not shown to simplify the diagrams.

Figure 3 shows the network initialized with all of the nodes **C1-C3** and **D1-D5** connected together to form a single main communication loop **80**. Other initial topologies are possible, including star, multiple rings, random connection, partially or completely disconnected, and the like.

Figure 4 shows the network configuration after the first disk controller **C1** has requested, and has been placed, on a first separate private communication loop **82**. At this point the first disk controller **C1** may issue an Open message.

Figure 5 shows the network configuration after the disk drive **D1** has been added to the first private communication loop **82**. The first disk controller **C1** and first disk drive **D1** can now exchange messages using the full bandwidth of the private communication loop **82** without any delays due to the other disk controllers **C2-C3** or disk drives **D2-D5**. If the first disk controller **C1** and the first disk drive **D1** close the message transfer operation, and the first private communication loop **82** times-out, the first disk controller **C1** and first disk drive **D1** are switched back to the main communication loop **80** returning the network to the configuration shown in Figure 3.

Figure 6 shows the network configuration resulting from Figure 5 after the first controller **C1** transmits an Open message containing the second disk drive **D2** as the destination address. Here the second disk drive **D2** has been added to the first private communication loop **82**. Now the first disk controller **C1** can communicate with both the first and second disk drives **D1-D2** without having to return to the main communication loop **80** to switch between the two disk drives **D1-D2**. The steps shown in Figure 5 and Figure 6 can be repeated to add more disk drives **D3-D5** to the first private communication loop **82**.

Figure 7 shows the network configuration resulting from Figure 6 after the first disk controller **C1** stops communicating with the first disk drive **D1** longer than then predetermined waiting period. After the first disk drive **D1** times-out, the switching hub **11** removes the first disk drive **D1** from the first private communication loop **82** and adds it back into the main communication loop **80**. This leaves the first disk controller **C1** and second disk drive **D2** with the entire bandwidth of the first private communication loop **82** to communicate.

Figure 8 shows the network configuration resulting from Figure 5 after the second disk controller **C2** has requested, and has been placed, on a second private communication loop **84**.

Figure 9 shows the network configuration resulting from Figure 8 after the second disk controller **C2** has transmitted an Open message containing the second disk drive **D2** as the destination address, causing the second disk drive **D2** to be switched from the main communication loop **80** to the second private communication loop **84**. Now both the first disk controller **C1** and second disk controller **C2** can communicate with their respective disk drives **D1-D2** simultaneously at the full bandwidth of the network medium. The steps shown in Figure 8 and Figure 9 can be repeated to create more private communication loops between other nodes in the network.

Figure 10 shows the network configuration resulting from Figure 8 after the second disk controller **C2** has transmitted an Open message containing the first disk drive **D1** as the destination address, and the first disk drive **D1** is already a part of the first private loop **82**. In this case, the switching hub **11** reconfigures the first private loop **82** to add the second disk controller **C2**, effectively eliminating the second private communication loop **84**. Now the first and second disk controllers **C1-C2** must arbitrate with each other for a share of the bandwidth of the first private communication loop **82** in order to communicate with the first disk drive **D1**. The steps shown in Figure 8 and Figure 10 can be repeated to add more disk controllers **C3** to the first private communication loop **82**.

While embodiments of the invention have been illustrated and described, it is not intended that these embodiments illustrate and describe all possible forms of the invention. Rather, the words used in the specification are words of description rather than limitation, and it is understood that various changes may be made without departing from the spirit and scope of the invention.